

**METHOD AND SYSTEM FOR A SINGLE-SIGN-ON MECHANISM WITHIN
APPLICATION SERVICE PROVIDER (ASP) AGGREGATION**

BACKGROUND OF THE INVENTION

5

1. Field of the Invention

The present invention relates to an improved data processing system and, in particular, to a method and apparatus for multicomputer data transferring. Still
10 more particularly, the present invention provides a method and apparatus for computer-to-computer authentication management.

2. Description of Related Art

15 In order to obtain operational efficiencies and reduce operating expenses, enterprises continue to adopt management practices that focus on core business opportunities while outsourcing nonessential functions, such as human resources, payroll, and accounting. One of
20 the many ways in which Internet-centric technologies have enhanced productivity is the facilitation of outsourced services for information technology (IT) management. Many Web-based Application Service Providers (ASPs) host, manage, and rent software applications at a central
25 location on an outsourced basis, thereby providing several benefits to the enterprise customers of the ASPs. In addition to reducing IT management costs, users can access the hosted applications through standard Internet connections and Web browsers, and users can easily
30 collaborate through a shared application even if separated by vast distances.

While ASPs can provide many benefits to enterprises, certain IT issues can continue to vex an enterprise even after outsourcing software applications. For example, in order for users to be presented with a coherent set of IT resources, an enterprise may need to host legacy applications that need to be integrated with the ASPs in some manner while also integrating the ASPs themselves. In addition, information security may also continue to be problematic, and security risks could actually increase by introducing an insecure infrastructure.

Moreover, corporate call centers have been found to spend a significant amount of resources resolving user password issues, and poor IT resource integration may actually increase user support costs because access management issues could become more complicated rather than more simplified with the introduction of an additional layer of security and an increased number of user identities and passwords.

To resolve some of these issues, several companies have extended the concept of an ASP to introduce an additional layer of software integration. While an ASP may rent a single "net-sourced" application to its subscribers, additional value can be added by providing common user registration, common billing, and a common login for a set of net-sourced applications based on an ASP infrastructure.

There are two major models for maintaining an ASP infrastructure. An ASP integrator hosts the net-sourced applications as well as the ASP infrastructure, and the hosted applications are written or modified to integrate with the ASP infrastructure, thereby greatly simplifying

the task of offering common services but also increasing the resource and management requirements of the ASP integrator. Alternatively, an ASP aggregator allows its subscribers to access multiple net-sourced applications that are hosted by a variety of ASPs; these ASPs may be independent third parties that host a single net-source application, or one or more of these ASPs may be an ASP integrator. In either case, the aggregated applications are typically not modified to use the infrastructure of the ASP aggregator. Instead, one or more adapters, typically created by the ASP aggregator, are used with the host applications to achieve the desired common services. Generally, aggregating net-sourced applications can reduce resource and management requirements while interfacing with a broader range of applications.

Single-sign-on is one of the key requirements of an ASP aggregator; otherwise, it may be difficult to enhance user productivity and to achieve reduced user support costs. Single-sign-on has been described as a fundamental goal in current and future implementations of access management applications; using a single-sign-on mechanism, a user can perform a single login operation for the purpose of authentication and can then access multiple applications thereafter without subsequent authentication operations. With respect to an ASP aggregator, a user can perform a single login operation to the ASP infrastructure and can access thereafter any of the multiple, aggregated, net-sourced applications during the same session.

While an ASP aggregator can enhance the efficiency of IT management, certain access management problems can still occur. Specifically, it may be assumed that an ASP aggregator maintains and controls a single-sign-on operation on behalf of a user, thereby presenting a common access mechanism that allows the user to access any application in a set of aggregated, hosted applications to which a user has subscribed. However, after the user successfully completes an authentication operation during a single-sign-on operation, the user is considered to be properly authenticated only during the particular ASP aggregator session that is associated with the single-sign-on operation.

Through ordinary operations of a client application, though, a user can save some form of session-specific information concerning an aggregated application during a session within the ASP infrastructure, i.e. after the user has been authenticated by the ASP aggregator and has been allowed to access a hosted application. After the session has expired, the user may try to use the saved session-specific information either by trying to re-establish a session directly with the aggregated application or by merely attempting to interact directly with the aggregated application at some later point in time. In either case, the user is presented with an error since the hosted application no longer recognizes the saved session-specific information as valid, generally because it is not valid across multiple sessions.

This type of access management problem occurs because the ASP aggregator model inherently presents a

form of middleware between the user and the host application. The single-sign-on mechanism is controlled and managed by the ASP aggregator, yet the aggregated applications are distinct entities that have some degree of independence from the ASP aggregator infrastructure. Hence, while the user is interacting directly with an aggregated application, a user has the ability to save some form of session-specific information concerning the aggregated application outside of the control of the ASP infrastructure.

Therefore, it would be advantageous to provide a method and a system for a single-sign-on mechanism within an ASP aggregator infrastructure that recovers from a scenario in which a user attempts to re-use saved session-specific information directly with a hosted application. It would be particularly advantageous if the method and the system maintained a coherent interface between the user and the ASP infrastructure.

SUMMARY OF THE INVENTION

The present invention is a method, system,
5 apparatus, or computer program product for providing a
single-sign-on mechanism within an ASP aggregator
service. An aggregator token is generated by an ASP
aggregator service and sent to a client device after its
10 user has been successfully authenticated during a
single-sign-on operation that is provided by the ASP
aggregator service. The aggregator token then
accompanies any request from the client to aggregated
applications within the ASP aggregator service's
15 infrastructure. The aggregator token comprises an
indication of an address or resource identifier within
the ASP aggregator service to which a client/user can be
redirected when the client/user needs to be authenticated
by the ASP aggregator service. In other words, the
20 address/identifier is associated with a logon resource;
when a request from a client is sent to this address, the
ASP aggregator service responds with an authentication
challenge to force the user to complete a single-sign-on
operation.

If an aggregated application (or an ASP that is
25 supporting the aggregated application) determines that
the client/user has not been properly authenticated and
should not receive access to the aggregated application
as requested within a request message that is received
from the client/user, then the aggregated application (or
30 its supporting ASP) can redirect the client/user to the
logon resource that is indicated by the aggregator token.

The redirectable message may itself include an address to which the ASP aggregator service should redirect the client/user after being authenticated so that the user may begin interacting with the desired application.

BRIEF DESCRIPTION OF THE DRAWINGS

5 The novel features believed characteristic of the invention are set forth in the appended claims. The invention itself, further objectives, and advantages thereof, will be best understood by reference to the following detailed description when read in conjunction
10 with the accompanying drawings, wherein:

Figure 1A depicts a typical network of data processing systems, each of which may implement the present invention;

Figure 1B illustrates a typical Web-based
15 environment in which the present invention may be implemented;

Figure 2A is a diagram that depicts a typical organization of entities with respect to an Application Service Provider (ASP) aggregator service;

20 **Figure 2B** is a diagram that depicts a typical ASP aggregator service and its relationships with clients and net-sourced applications in more detail than in **Figure 2A**;

25 **Figure 3A** is a diagram that depicts a typical graphical user interface (GUI) window that may be presented to a user by a browser application;

Figure 3B is a diagram that depicts a typical browser application window after accessing a Web site for an ASP aggregator service;

Figure 3C is a diagram that depicts a typical browser application window after successfully authenticating to an ASP aggregator service;

Figure 3D is a diagram that depicts a typical browser application window after a user has requested to access an aggregated application;

Figure 3E is a temporal flow diagram that depicts some of the actions and communication traffic for a typical single-sign-on operation with a typical ASP aggregator service;

Figure 4 is a temporal flow diagram that depicts some of the actions and communication traffic for a single-sign-on operation with an ASP aggregator service that is implemented in accordance with the present invention; and

Figures 5A-5B are temporal flow diagrams that depict some of the actions and communication traffic for a scenario in which an ASP aggregator service's infrastructure employs an aggregator token in accordance with the present invention to recover from a user's attempted interaction with an aggregated application without the user having been previously authenticated through the ASP aggregator service.

DETAILED DESCRIPTION OF THE INVENTION

The present invention is directed to a system and a methodology for secure client-server access management.

5 Computers that implement the present invention may be dispersed throughout a network or distributed data processing system. As background, a typical organization of hardware and software components within a distributed data processing system is described prior to describing

10 the present invention in more detail.

With reference now to the figures, **Figure 1A** depicts a typical network of data processing systems, each of which may contain and/or operate the present invention. Distributed data processing system **100** contains network

15 **101**, which is a medium that may be used to provide communications links between various devices and computers connected together within distributed data processing system **100**. Network **101** may include permanent connections, such as wire or fiber optic cables, or

20 temporary connections made through telephone or wireless communications. In the depicted example, server **102** and server **103** are connected to network **101** along with storage unit **104**. In addition, clients **105-107** also are connected to network **101**. Clients **105-107** and servers **102-103** may

25 be represented by a variety of computing devices, such as mainframes, personal computers, personal digital assistants (PDAs), etc. Distributed data processing system **100** may include additional servers, clients, routers, other devices, and peer-to-peer architectures

30 that are not shown.

In the depicted example, distributed data processing system 100 may include the Internet with network 101 representing a worldwide collection of networks and gateways that use various protocols to communicate with one another, such as Lightweight Directory Access Protocol (LDAP), Transport Control Protocol/Internet Protocol (TCP/IP), Hypertext Transport Protocol (HTTP), Wireless Application Protocol (WAP), etc. Of course, distributed data processing system 100 may also include a number of different types of networks, such as, for example, an intranet, a local area network (LAN), or a wide area network (WAN). For example, server 102 directly supports client 109 and network 110, which incorporates wireless communication links. Network-enabled phone 111 connects to network 110 through wireless link 112, and PDA 113 connects to network 110 through wireless link 114. Phone 111 and PDA 113 can also directly transfer data between themselves across wireless link 115 using an appropriate technology, such as Bluetooth™ wireless technology, to create so-called personal area networks (PAN) or personal ad-hoc networks. In a similar manner, PDA 113 can transfer data to PDA 107 via wireless communication link 116.

The present invention could be implemented on a variety of hardware platforms; **Figure 1A** is intended as an example of a heterogeneous computing environment and not as an architectural limitation for the present invention.

With reference now to **Figure 1B**, a diagram depicts a typical computer architecture of a data processing system, such as those shown in **Figure 1A**, in which the present

invention may be implemented. Data processing system 120 contains one or more central processing units (CPUs) 122 connected to internal system bus 123, which interconnects random access memory (RAM) 124, read-only memory 126, and input/output adapter 128, which supports various I/O devices, such as printer 130, disk units 132, or other devices not shown, such as a audio output system, etc. System bus 123 also connects communication adapter 134 that provides access to communication link 136. User interface adapter 148 connects various user devices, such as keyboard 140 and mouse 142, or other devices not shown, such as a touch screen, stylus, microphone, etc. Display adapter 144 connects system bus 123 to display device 146.

Those of ordinary skill in the art will appreciate that the hardware in **Figure 1B** may vary depending on the system implementation. For example, the system may have one or more processors, including a digital signal processor (DSP) and other types of special purpose processors, and one or more types of volatile and non-volatile memory. Other peripheral devices may be used in addition to or in place of the hardware depicted in **Figure 1B**. The depicted examples are not meant to imply architectural limitations with respect to the present invention.

In addition to being able to be implemented on a variety of hardware platforms, the present invention may be implemented in a variety of software environments. A typical operating system may be used to control program execution within each data processing system. For

example, one device may run a Unix® operating system, while another device contains a simple Java® runtime environment. A representative computer platform may include a browser, which is a well known software application for accessing documents and processing messages in a variety of formats, such as graphic files, word processing files, Extensible Markup Language (XML), Hypertext Markup Language (HTML), Handheld Device Markup Language (HDML), Wireless Markup Language (WML), Security Assertion Markup Language (SAML), and various other formats and types of files.

Given the background information provided above, a typical ASP aggregator service is described in association with the following **Figures 2A-2B** and **Figures 3A-3E**; a typical ASP aggregator service may be implemented in a variety of forms and models in accordance with open standards or proprietary technology. After discussing a typical ASP aggregator service, the present invention will be described in more detail.

With reference now to **Figure 2A**, a typical organization of entities is shown with respect to an Application Service Provider (ASP) aggregator service. Clients **202-203** interact with ASP aggregator service **205**, which provides an aggregation service for ASP applications **210-212**. ASP applications **210-212** are applications that are hosted on a central server and accessed by clients **202-203** through a network, such as network **101** shown in **Figure 1**; hence, applications **210-212** may be described as "hosted applications" or "net-sourced applications" to distinguish them from applications that reside at a client. Additionally, in

an attempt to increase efficiencies and productivity, ASP aggregator service 205 centralizes access management for applications 210-212 by providing a single authentication gateway for users that want to access the applications.

5 Hence, applications 210-212 may also be described as "aggregated applications". It may be assumed that ASP aggregator service 205 provides some form of single-sign-on functionality.

10 With reference now to **Figure 2B**, a typical ASP aggregator service and its relationships with clients and net-sourced applications are shown in more detail than in **Figure 2A**. **Figure 2B** illustrates the variety of application configurations that may be supported by a typical ASP aggregator service. Because the present invention may be implemented into the infrastructure of a typical ASP aggregator service, the present invention may also support a wide variety of application configurations. In order to provide background information for the variety of contexts in which the present invention may be implemented, a subset of these configurations are described below. However, the depicted examples are not meant to imply architectural or functional limitations with respect to the present invention.

25 Client 251 supports browser application 252, which maintains temporary cookie cache 253 and permanent cookie database or file 254 for storing cookies that have been received from servers, as is well-known in the art. In order to depict a more generalized client application environment, **Figure 2B** shows client 255 supporting

30

net-enabled application 256 that may be used by a user of client 255 to perform functional operations through a connected network. Net-enabled application 255 itself may be temporary by being downloaded through a connected network. Net-enabled application 256 may store tokens, such as authentication tokens that have been received from servers, in token database 257.

Via network 260, client 251 or client 255 may access ASP aggregator service 261. It may be assumed that a user of a client device has already established some type of account with the ASP aggregator service, e.g., by registering with the ASP aggregator service and paying a subscription fee, after which the ASP aggregator service has generated a user identity for the user and appropriate credentials for authenticating the user's identity when the user presents a request for access to resources protected by ASP aggregator service 261.

In this example, ASP aggregator service 261 uses aggregator subscriber database 262 to store user identifier information 263, which is associated with subscription list 264 and application credentials 265. Subscription list 264 indicates the set of applications to which a user has previously requested to access through the aggregator service, i.e. the set of aggregated applications to which the user has subscribed. Each of the aggregated applications may require an application credential 265 that the ASP aggregator must present to an aggregated application when attempting to complete an authentication operation with the aggregated application on behalf of the user.

When a user logs into the ASP aggregator service, the user is authenticated with respect to the ASP aggregator service using the aggregator service authentication unit 266. After a user successfully
5 completes the ASP aggregator service's authentication operation, the ASP aggregator service attempts to authenticate the user with respect to each of the user's subscribed applications as necessary.

ASP aggregator service 261 may aggregate a variety
10 of net-sourced applications, as briefly mentioned with respect to **Figure 2A**. In **Figure 2B**, the aggregated applications are shown as being accessible through network 260; these applications may be distributed
15 throughout the network and do not necessarily have to be physically located on servers that are operated by the ASP aggregator service. However, these application must have an appropriate interface that allows the ASP aggregator service to communicate with them or their
20 supporting frameworks such that the ASP aggregator service can perform authentication operations with the applications on behalf of customers of the ASP aggregator service.

In the exemplary organization shown in **Figure 2B**, aggregator-hosted applications 271 and 272 may be
25 supported on a server that is physically maintained by the ASP aggregator service; these applications are shown as being generically accessed through network 260 because the applications may not be physically co-located with servers that support the majority of the functionality of
30 ASP aggregator service 261. In the case of application 271, adapter 273 has been written and interfaced with

application 271 to provide the necessary functionality that allows the ASP aggregator service to perform authentication functions on behalf of a user. In contrast, application 272 may have been originally implemented in accordance with open standards that allow the ASP aggregator service to perform its desired single-sign-on operations.

In a manner similar to aggregator-hosted applications 271-272, ASP-hosted applications 274-275 may also be accessible through network 260 and may also be distinguishable by the fact that application 274 already has the necessary authentication functionality whereas application 275 must be accessed through adapter 276. However, applications 274-275 may be supported on servers that are physically maintained by one or more ASPs. In other words, a user could access net-sourced applications 274-275 on a subscription basis independently through one or more ASPs without the agency of ASP aggregator service 261.

Legacy application 277 may be supported on a server that is physically maintained on a server that is operated by a customer of the ASP aggregator service. For example, the customer may not have desired (or may not have been able) to relinquish control over a legacy application by porting it to a server operated by ASP aggregator service 261. In this example, however, the customer has incorporated single-sign-on functionality with the legacy application through the ASP aggregator service by interfacing adapter 278 with the legacy application for that purpose.

Using the environment shown in **Figure 2B**, the users of the client devices may be given the impression that they are only interacting with the ASP aggregator service even though the user may be interacting with a variety of applications that are supported on multiple, distinct, physically dispersed servers. For example, using browser application 252, a user may select or enter a Uniform Resource Locator (URL), or more generally, a Uniform Resource Identifier (URI), that directs the browser application to the ASP aggregator service. After that point in time, the user does not need to be cognizant of the Domain Name System (DNS) addresses that are used to present an integrated interface to the user's subscribed, aggregated applications. **Figures 3A-3D** show some of the typical Web pages that may be presented to a user within a typical browser application while using the single-sign-on functionality of a typical ASP aggregator application.

With reference now to **Figure 3A**, a diagram shows a typical graphical user interface (GUI) window that may be presented to a user by a browser application. Window 300 contains GUI controls 302 for allowing a user to navigate the World Wide Web. Window 300 also displays the content of a retrieved Web page, as identified by the Web page's URL 304, within content area 306.

With reference now to **Figure 3B**, a diagram shows a typical browser application window after accessing a Web site for an ASP aggregator service. Window 300 contains URL 310 that identifies a login page from an ASP aggregator service's Web site. The content area of

window 300 contains an HTML form that a user may use to submit user identifier 312 and user password 314 by selecting "LOGON" button 316; the user identifier and password are used by the ASP aggregator service to authenticate the user, after which the single-sign-on functionality of the ASP aggregator service performs any subsequent authentication operations with aggregated applications on behalf of the user. Other information, such as the user's corporation, division, etc., may also be required to be entered as necessary in order to authenticate the user.

With reference now to **Figure 3C**, a diagram shows a typical browser application window after successfully authenticating to an ASP aggregator service. Window 300 contains URL 320 that identifies an application workspace page that has been received and displayed by the browser application; this particular Web page may have been dynamically generated in response to the user's successful authentication by the ASP aggregator service. After determining the applications for which the user has an active subscription, hyperlinks 321-323 to those applications have been inserted into the user's application workspace page.

With reference now to **Figure 3D**, a diagram shows a typical browser application window after a user has requested to access an aggregated application. Window 300 contains URL 330 that identifies a main page for the aggregated application that was chosen by the user by selecting hyperlink 323 within the Web page that is shown in **Figure 3C**. Since the user has already been

authenticated by the ASP aggregator service, the user does not need to perform another authentication process with the selected application because the ASP aggregator service performs any subsequent authentication processes on behalf of the user as part of its single-sign-on functionality. The manner in which the user's client and the servers of the ASP aggregator service infrastructure interact is shown in more detail in **Figure 3E**.

With reference now to **Figure 3E**, a temporal flow diagram shows some of the actions and communication traffic for a typical single-sign-on operation with a typical ASP aggregator service. The actions and operations shown in **Figure 3E** depict the underlying processing that may be necessary in order for the user to be presented with the Web pages shown in **Figures 3A-3D**. Although the following examples employ a browser application, other types of client applications that interact with an ASP aggregator service may also be used, as mentioned above with respect to **Figure 2B**. In addition, various document formats and communication protocols may be used as is well-known in the art.

A user of client device 350 desires to access an aggregated application via an ASP aggregator service that is supported by server 352. In one of a variety of well-known manners, the user directs a browser application to a Web page that is available from the ASP aggregator service (step 354), which causes a Web page request to be sent to the ASP aggregator service (step 356), most likely in the form of an HTTP Request message. More generally, the user may request some type of

computational resource that is protected by the ASP aggregator service.

The ASP aggregator service determines that the user/client has not yet been authenticated with the ASP aggregator service (step 358) and generates an authentication challenge, e.g., a logon page (step 360), after which the logon page is sent to the client (step 362), most likely in the form of an HTML form document within an HTTP Response message.

After the user's browser application has displayed the HTML form document, the user can enter authentication data (step 364) and select a control to submit the authentication data (step 366). After the ASP aggregator service has authenticated the user (step 368), an application workspace page is generated (step 370) and returned to the client (step 372). After selecting a hyperlink that represents the user's choice of an aggregated application to be used (step 374), an application request message is returned to the ASP aggregator service (step 376), which then determines the selected application from the requested URL that is returned in the HTTP Request message. The ASP aggregator service then generates an application authentication token (AAT) that is appropriate for the user and the selected application (step 378). The client's request is then modified to include the application authentication token in an appropriate manner (step 380), and the ASP aggregator service returns the modified request as an HTTP Redirect message that is directed to a specific URL

that is associated with the user's requested application (step 382).

Assuming that the HTTP protocol is being used for communication between the entities shown in **Figure 3E**, the ASP aggregator service's server may use an HTTP Redirect message. An HTTP Redirect allows a server to respond to a client request with instructions to load a resource at a different location, and in response, most browsers will automatically request the new resource in response to a redirect. When the browser receives the HTTP Redirect, the browser issues a new HTTP Request using the redirected URI or URL provided in the HTTP Redirect.

In response to receiving the modified request, the client automatically redirects the request to the indicated URL (step 384) by sending an appropriate HTTP message through the network (step 386). The request is routed to server 388 that receives data traffic for the indicated URL, which in this case is also a server that supports, in some manner, the aggregated application requested by the user. The application authentication token that was placed into the modified request is then verified (step 390). Assuming that the verification process is successful, then a response is generated (step 392) and returned to the client (step 394). The response would include some type of HTML document that is presented to the user by the client's browser application, and the user can then interact with the selected, aggregated application through a series of GUI actions as is well-known in the art (step 396).

It should be noted that a typical ASP aggregator may use a variety of security techniques to ensure the integrity of its operations and its communications. For example, when the ASP aggregator service sends the initial logon page to the client, shown as step 362 in **Figure 3E**, the server at the ASP aggregator service and the client may first establish a secure HTTPS session in which subsequent communications between the client and the server are encrypted. Assuming that the logon operation is being implemented in a lightweight manner, the logon request may be a simple request to access a markup language document that represents a login Web page. In this manner, the client does not require a dedicated logon application to have been previously installed, and the ASP aggregator service can rely upon standard functionality in a typical browser application on the client machine to support the logon process. In an alternative implementation, the ASP aggregator service could send a logon applet to the client, and the logon applet could comprise cryptographic functionality to handle a message authentication code (MAC) from the client to the server. These encryption techniques are well-known in the art and have not been shown in the figures.

Moreover, it should be noted that the ASP aggregator service shown in the figures may employ a variety of models for its security operations. For example, a typical pull model allows an ASP, i.e. an entity that is hosting a net-sourced application, to pull authentication information from the ASP aggregator based on references or tokens that have been presented by a client to the

ASP. Alternatively, a typical push model allows the ASP aggregator to push authentication information to an ASP such that the ASP may already possess any necessary authentication information that is associated with references or tokens that are subsequently presented by a client to the ASP. As another alternative, the use of an ASP aggregator between a user and an ASP does not preclude the use of a third-party security service for handling credentials in some manner.

It should also be noted that the ASP aggregator service shown in the figures may use a variety of mechanisms for storing and communicating tokens among the clients and servers. The protocols for communicating security information, which may include tokens, may correspond with a particular security model in the set of security models that were mentioned above. Messages may be formatted in accordance with Security Assertion Markup Language (SAML), Security Service Markup Language (so-called "S2ML"), Information Technology Markup Language (ITML), or other markup languages or formats as necessary.

The data format of any tokens may vary depending upon system implementation in accordance with proprietary or standard formats. For example, a token may also include an optional timestamp for publicly indicating and limiting the useful lifetime of the token. A token may be formatted as a binary string, as an encoded ASCII string, or in some other interpretable format. A token may be optionally formatted in accordance with various standards, such as PKCS (Public Key Cryptography Standards) specifications for enveloped data. In other

words, any information within a token may be encrypted to hide the information so as to limit the risk that it might be misappropriated. It should be noted either that the entire token can be an encrypted data item or that individual data items can be encrypted and then placed within the token.

As is well-known, tokens may be implemented in a variety of forms. For example, a token may be associated with a user or client by appending a string that represents the token to a URL (or URI) string within a message from a server, so-called "URL stuffing". When a receiving entity examines the URL string, the portion of the URL string that represents the token can easily be separated from the portion of the URL string that identifies the resource that is being requested.

Alternatively, tokens may be implemented as cookies. As is well known, a cookie is a data item that is stored on a client by a server through a particular user's web browser. When a user of a client machine visits a Web server, the server may return a cookie to the user's browser to be stored in a client-side cookie cache. When a cookie is "set", i.e. stored, as part of an HTTP transaction, it may include the path, i.e. domain, for which the cookie is valid, the cookie's name and value, and other optional attributes, such as the cookie's expiration date. In most cases, the client browser automatically stores the cookie data by default, sometimes without giving the user the option or the knowledge of it being done. When the client/user "revisits" a domain that was associated with the cookie, the cookie is automatically sent with any messages that

are addressed to the domain, thereby identifying the client/user to the server that supports the domain. Generally, possession of a cookie is not equated with proof of identity for various security reasons. Cookies
5 can be either persistent cookies, which are stored on disk and persist between browser sessions, or memory cookies, which are stored in volatile memory and are active only for a current browser session. Persistent cookies typically reside in a client file, such as
10 "cookies.txt", after the browser application has been terminated and are available during the next browser session.

It should be understood by one of ordinary skill in the art that user and client are relatively
15 interchangeable. For example, in some instances, the user may initiate a process, and in other instances, a client may automatically initiate a process on behalf of the user. When a user is described as being the destination or origination of some form of data, the
20 client receives and sends the data on behalf of the user. Hence, a cookie or a token can be described as being associated with a client or with a user.

During interaction with the aggregated application, shown as step 396 in **Figure 3E**, the user may save
25 session-specific information or information about the aggregated application which is later presented by the user to the application or which allows the user to attempt to interact with the application directly. For example, the user may save a URL for a Web page that was
30 displayed while the user was interacting with the application; typically, browser applications have a

"bookmark" operation that conveniently allows a user to save the URL of a Web page. In the context of using the ASP aggregator service to implement single-sign-on functionality, a previously authenticated user may use a bookmarked URL from a Web page associated with an aggregated application while the user's current session is active.

At some point in time, however, the user's session will be terminated, e.g., by a logout operation or by closing the client browser application. Thereafter, the user may restart the browser application, and any subsequent attempt by the user to interact with the aggregated application will fail. For example, if the user attempts to access the aggregated application through the bookmarked URL, even if the bookmarked URL comprises a previously valid application authentication token, then an error will occur because the user's previous session has ended; the application authentication token that is used to identify an authenticated user, shown being returned in step 382 in **Figure 3E**, will generally be session-based and, therefore, not valid between sessions. If the user attempts to interact directly with the aggregated application in some other manner, such as by directing a browser application to the aggregated application by entering a URL or a portion of a URL that identifies the aggregated application in some manner, then an error will also occur because the request will not be accompanied by any instance of an application authentication token. In other words, the user will not be able to interact directly with the aggregated application until the user

has completed another authentication operation with the ASP aggregator service.

In order to solve the problem in which a user attempts to interact directly with an ASP without using an ASP aggregator service as an intermediate agent, the present invention offers a dynamic identification of the ASP aggregator service to which the user subscribes so that the user can be directed to complete the single-sign-on operation at the ASP aggregator service. In this manner, the present invention is intended to be able to be integrated into a variety of configurations for ASP aggregator services in order to solve the above-identified problem among ASP aggregator services, as discussed in more detail below.

The present invention introduces an aggregator token that is returned to the client/user by the ASP aggregator service in response to a successful, initial, authentication operation. When the ASP aggregator service generates an aggregator token for the client/user, an appropriate address is included in the aggregator token. The address in the aggregator token identifies a logon application, a logon start page, or similar logon resource to which the client/user needs to be directed in order to complete the authentication operation that is provided as part of the single-sign-on functionality of the ASP aggregator service.

The client stores the aggregator token and forwards the aggregator token with each request to the ASP aggregator service's infrastructure, including the ASPs that have been aggregated into the ASP aggregator service. The aggregator token is also sent to an ASP

along with any request message that is sent to the ASP while the user is attempting to interact with an aggregated application that is supported by the ASP. Hence, even if a user attempts to interact directly with an ASP in some manner without using the single-sign-on feature of the ASP aggregator service, i.e. without having been authenticated by the ASP aggregator service, then the ASP can examine the aggregator token and determine the ASP aggregator service to which the client/user should be redirected in order to complete the single-sign-on process at the ASP aggregator service. After the client/user has completed the authentication operation at the ASP aggregator service, then the client/user can be redirected back to the originating ASP so that the user can interact with the desired aggregated application.

With reference now to **Figure 4**, a temporal flow diagram shows some of the actions and communication traffic for a single-sign-on operation with an ASP aggregator service that is implemented in accordance with the present invention. **Figure 4** depicts actions and communication traffic for an ASP aggregator service's infrastructure that uses aggregator tokens; **Figures 5A-5B** then depict some of the actions and communication traffic for a scenario in which an ASP aggregator service's infrastructure employs an aggregator token to recover from a user's attempted interaction with an aggregated application without the user having been previously authenticated through the ASP aggregator service. It should be noted again that the following examples describe an embodiment of the present invention using

cookies, HTTP, Web documents, and related standards and protocols, but the depicted examples are not meant to imply architectural or functional limitations with respect to the present invention.

5 A user of client device 400 desires to access an aggregated application via an ASP aggregator service that is supported by server 402. The user directs a browser application to a Web page that is available from the ASP aggregator service (step 404), which causes a Web page
10 request to be sent to the ASP aggregator service in an HTTP Request message (step 406).

The ASP aggregator service determines that the user/client has not yet been authenticated with the ASP aggregator service (step 408) and generates an
15 authentication challenge page, i.e. logon page (step 410), after which the logon page is sent to the client as an HTML form document within an HTTP Response message (step 412).

After the user's browser application has displayed
20 the logon page, the user can enter authentication data (step 414) and select a control to submit the authentication data (step 416). After the ASP aggregator service has authenticated the user (step 418), the ASP aggregator service generates an aggregator token (step
25 420).

The aggregator token comprises an address that indicates the logon resource to which a user should be redirected if an ASP, aggregated application, or other entity in the ASP aggregator service's infrastructure
30 determines that the user has not been properly

authenticated when processing a request from the user for access to a resource that is supported or protected by the entity that received the request. The specific content of the logon resource indicator may depend upon a variety of factors, such as the protocols that are supported by the ASP aggregator service. Moreover, the logon resource indicator may depend upon the identity of the user; a first set of users may be directed to a first logon resource and should be associated with a first logon resource identifier, whereas a second set of user may be directed to a second logon resource and should be associated with a second logon resource identifier. In the example shown in **Figure 4**, the aggregator token is an HTTP cookie comprising a URL of a logon page to which the user should be redirected if the user has not been properly authenticated within the ASP aggregator service's infrastructure; the cookie can be assumed to be associated with a global domain that includes any addresses within the ASP aggregator service's infrastructure.

The ASP aggregator service generates a response that includes the aggregator token and an application workspace page that is specifically tailored for the user (step 422), and the application workspace page is returned to the client (step 424). The client automatically stores the aggregator token in an appropriate manner, such as a permanent cookie cache or token database (step 426). After selecting a hyperlink that represents the user's choice of an aggregated application to be used (step 428), an application request message is returned to the ASP aggregator service (step

430), which then determines the selected application from the requested URL that is returned in the HTTP Request message. The ASP aggregator service then generates an application authentication token (AAT) that is
5 appropriate for the user and the selected application (step 432). The client's request is then modified to include the application authentication token in an appropriate manner (step 434), and the ASP aggregator service returns the modified request as an HTTP Redirect
10 message that is directed to a specific URL that is associated with the user's requested application (step 436).

In response to receiving the modified request, the client automatically redirects the request to the
15 indicated URL (step 438) by sending an appropriate HTTP message through the network (step 440). The request is routed to server 442 that receives data traffic for the indicated URL, which in this case is also a server that supports, in some manner, the aggregated application
20 requested by the user. The application authentication token that was placed into the modified request is then verified (step 444). Assuming that the verification process is successful, then a response is generated (step 446) and returned to the client (step 448). The response
25 would include some type of HTML document that is presented to the user by the client's browser application, and the user can then interact with the selected, aggregated application (step 450).

With reference now to **Figures 5A-5B**, temporal flow
30 diagrams depict some of the actions and communication

traffic for a scenario in which an ASP aggregator service's infrastructure employs an aggregator token in accordance with the present invention to recover from a user's attempted interaction with an aggregated application without the user having been previously authenticated through the ASP aggregator service.

Referring now to **Figure 5A**, client device 502 interacts with server 504 that supports an ASP or an aggregated application, similar to the manner shown in **Figure 4**; in response to user actions (step 506), an application can generate responses (step 508), and the requests and responses are depicted as communication traffic 510. During the initial authentication process, an aggregator token would be stored at the client and then subsequently sent along with requests from the client when appropriate. At some point in time during the user interactions, the user saves some type of session-specific information (step 512); for example, the user may bookmark a URL that is associated with an aggregated application.

Eventually, the user's session will expire (step 514), e.g., when the user closes the browser session. At that point in time, the user has severed the association between the user's active session and the user's previous authentication.

At some later point in time, the user can start a new session (step 516), e.g., by starting a new browser session, after which the user may attempt to use the previously saved session-specific information (step 518). For example, the user may attempt to access an aggregated

application using a bookmarked URL, which causes the user's browser application to send a request to the specified URL (step 520), and the aggregator token is automatically sent along with the request. After the aggregated application receives the request, it will determine that the request is not accompanied by a valid application authentication token (step 522), and the user is denied access to the application.

Rather than return an error to the user, the aggregator token is examined to determine a logon resource at the ASP aggregator service, such as a URL for a logon page (step 524). After obtaining the URL (or the URI) for the logon resource, the incoming request message (or a newly generated message) can be modified to redirect the client to the logon resource (step 526), thereby causing the client to be forced to complete an authentication operation at the ASP aggregator service. In addition, the message includes the return URL of the aggregated application so that the ASP aggregator service can redirect the client back to the aggregated application after the user has been authenticated.

The functionality for examining the aggregator token does not necessarily need to be incorporated directly into the aggregated application. The aggregated application may be supported by an ASP in which the ASP operates its own servers and infrastructure. An adapter that is associated with the aggregated application can be written or modified to include functionality for checking whether an incoming request has an associated aggregator token and then examining the aggregator token for the address or identifier of the appropriate logon resource.

The redirect message is sent to the client (step 528), and the client automatically redirects the request to the ASP aggregator service as indicated by the URL of the logon resource (step 530). Although it is not necessarily needed by the ASP aggregator service at this stage, the redirected request would preferably be accompanied automatically by the aggregator token; assuming that the aggregator token is implemented as an HTTP cookie, the aggregator token will be automatically sent to the ASP aggregator service.

Continuing with **Figure 5B**, redirected request 532 is received by server 534 that is supporting the ASP aggregator service, which determines that the client/user is not properly authenticated (step 536). In a manner similar to that shown in **Figure 4**, the ASP aggregator service generates an authentication challenge page, e.g. logon page (step 538), after which the logon page is sent to the client (step 540).

After the user's browser application has displayed the logon page, the user can enter authentication data (step 542) and select a control to submit the authentication data (step 544). After the ASP aggregator service has authenticated the user (step 546), the ASP aggregator service preferably generates a new aggregator token (step 548) to be associated with the client/user; alternatively, if the redirected request that is being processed was accompanied by an aggregator token, then a new aggregator token would not necessarily need to be generated because the client is already storing an aggregator token.

The ASP aggregator service then determines the original aggregated application from which the redirected request was received by examining the original address or URL that was included in the redirected request, and the ASP aggregator service generates an application authentication token (AAT) that is appropriate for the user and the aggregated application (step 550). The client's request is then modified to include the newly generated aggregator token and the newly generated application authentication token in an appropriate manner (step 552), and the ASP aggregator service returns the modified request as a message that is to be directed to the user's originally requested application (step 554).

In response to receiving the modified request, the client automatically stores the newly generated aggregator token in an appropriate manner, such as a permanent cookie cache or token database (step 556). The client then automatically redirects the request to the indicated URL or address (step 558) by sending an appropriate message (step 560). The request is routed to server 504 that receives data traffic for the aggregated application that was originally requested by the user. The application authentication token that was placed into the modified request is then verified (step 562).

Assuming that the verification process is successful, then a response is generated (step 564) and returned to the client (step 566). The response would include some type of HTML document that is presented to the user by the client's browser application, and the user can then interact with the aggregated application (step 568).

Figures 5A-5B depict a scenario in which an ASP aggregator's infrastructure recovers from a user's attempted interaction with an aggregated application without the user having been previously authenticated through the ASP aggregator service. It should be noted that a slightly different scenario is possible, which is described below.

In this scenario, the user may have been authenticated by the ASP aggregator and may have interacted with a first aggregated application. At some later point in time, in a manner similar to that described above with respect to **Figures 5A-5B**, the user may attempt to interact directly with a second aggregated application using some type of saved session-specific information, e.g., a bookmarked URL that is associated with the second aggregated application. Since the user has attempted to interact directly with the aggregated application without the intermediate step of using the application workspace page, the second aggregated application would not receive an application authentication token along with the client/user request. The second aggregated application would receive and examine the aggregator token, after which the client/user would be redirected to the ASP aggregator.

In contrast to the scenario described with respect to **Figures 5A-5B**, since the client/user has been previously authenticated by the ASP aggregator, the client/user would not receive an authentication challenge from the ASP aggregator. In other words, steps 538-546 in **Figure 5B** would be unnecessary. Because the ASP aggregator has already authenticated the client/user, the

ASP aggregator would immediately generate the application authentication token that is needed by the client/user with respect to the second aggregated application and then redirect the client/user to the second aggregated application. After the second aggregated application is received and verified, the user may interact with the second aggregated application. Hence, in this scenario, the client's request to the second aggregated application undergoes two redirections, i.e. to and from the ASP aggregator and the second aggregated application, in a manner that should be transparent to the user. In this scenario, the aggregator token enables the user to perform a single sign-on operation yet still jump to direct interaction with aggregated applications without using an application workspace page.

The advantages of the present invention should be apparent in view of the detailed description of the invention that is provided above. The present invention requires minimal modification to an ASP aggregator service's infrastructure. The ASP aggregator service needs to be able to generate an aggregator token, and an aggregated application or its supporting ASP needs to be able to use the aggregator token when necessary. A typical ASP aggregator service's infrastructure can be easily modified to include the functionality of the present invention.

An aggregator token is generated by the ASP aggregator service and sent to the client device of a user that has subscribed to the ASP aggregator service after the user has successfully completed the authentication portion of the single-sign-on feature that

is provided by the ASP aggregator service. The aggregator token then accompanies any request from the client to aggregated applications within the ASP aggregator service's infrastructure. The aggregator token comprises an indication of an address or resource identifier within the ASP aggregator service to which a client/user should be redirected to complete the authentication portion of the single-sign-on feature that is provided by the ASP aggregator service. In other words, the address/identifier is associated with a logon resource; when a request from a client is sent to this address, the ASP aggregator service responds with an authentication challenge to force the user to complete a single-sign-on operation.

If an aggregated application (or an ASP that is supporting the aggregated application) determines that the client/user has not been properly authenticated and should not receive access to the aggregated application as requested within a request message that is received from the client/user, then the aggregated application (or its supporting ASP) can redirect the client/user to the logon resource that is indicated by the aggregator token. The redirectable message may itself include an address to which the ASP aggregator service should redirect the client/user after being authenticated so that the user may begin interacting with the desired application.

When the present invention is implemented, a user who attempts to use an aggregated application without being properly authenticated is presented with a logon page rather than an error page. After submitting authentication data, the user is then presented with the

desired application. In this manner, the user is not burdened with understanding the operation of the ASP aggregator service's infrastructure. From one perspective, the aggregator token enables a valid authentication session for the user to be "jump started".

Moreover, an aggregated application or its supporting ASP may be aggregated into multiple ASP aggregator services, each of which may have a unique address for a logon resource. When it is determined that a client or user has not been properly authenticated, then the aggregated application or its supporting ASP needs to be able to determine the appropriate ASP aggregator service to which the client/user needs to be redirected. The aggregator token allows this determination to be performed in a dynamic manner; all of the logic for associating the subscribed users with specific logon resources remains with the ASP aggregator service.

It is important to note that while the present invention has been described in the context of a fully functioning data processing system, those of ordinary skill in the art will appreciate that the processes of the present invention are capable of being distributed in the form of instructions in a computer readable medium and a variety of other forms, regardless of the particular type of signal bearing media actually used to carry out the distribution. Examples of computer readable media include media such as EPROM, ROM, tape, paper, floppy disc, hard disk drive, RAM, and CD-ROMs and transmission-type media, such as digital and analog communications links.

The description of the present invention has been presented for purposes of illustration but is not intended to be exhaustive or limited to the disclosed embodiments. Many modifications and variations will be apparent to those of ordinary skill in the art. The
5 embodiments were chosen to explain the principles of the invention and its practical applications and to enable others of ordinary skill in the art to understand the invention in order to implement various embodiments with
10 various modifications as might be suited to other contemplated uses.

11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
221